Artificial neural networks training acceleration through network science strategies

Lucia Cavallaro, Ovidiu Bagdasar, Pasquale De Meo, Giacomo Fiumara & Antonio Liotta

Soft Computing
A Fusion of Foundations,
Methodologies and Applications

ISSN 1432-7643

Soft Comput
DOI 10.1007/s00500-020-05302-y

Springer

Soft Computing
https://doi.org/10.1007/s00500-020-05302-y

FOCUS

Artificial neural networks training acceleration through network science strategies

Lucia Cavallaro[1] · Ovidiu Bagdasar[1] · Pasquale De Meo[2] · Giacomo Fiumara[3] · Antonio Liotta[4]

Abstract
The development of deep learning has led to a dramatic increase in the number of applications of artificial intelligence. However, the training of deeper neural networks for stable and accurate models translates into artificial neural networks (ANNs) that become unmanageable as the number of features increases. This work extends our earlier study where we explored the acceleration effects obtained by enforcing, in turn, scale freeness, small worldness, and sparsity during the ANN training process. The efficiency of that approach was confirmed by recent studies (conducted independently) where a million-node ANN was trained on non-specialized laptops. Encouraged by those results, our study is now focused on some tunable parameters, to pursue a further acceleration effect. We show that, although optimal parameter tuning is unfeasible, due to the high non-linearity of ANN problems, we can actually come up with a set of useful guidelines that lead to speed-ups in practical cases. We find that significant reductions in execution time can generally be achieved by setting the revised fraction parameter (ζ) to relatively low values.

Keywords Network science · Artificial neural networks · Multilayer perceptron · Revise phase

1 Introduction

The effort to simulate the human brain behaviour is one of the top scientific trends today. In particular, deep learning strategies pave the way to many new applications, thanks to their ability to manage complex architectures. Notable examples are: speech recognition (Hinton et al. 2012), cyber-security (Berman et al. 2019), image (Krizhevsky et al. 2017), and signal processing (Dong and Li 2011). Other applications gaining popularity are related to bio-medicine (Cao et al. 2018) and drug discovery (Chen et al. 2018; Ruano-Ordás et al. 2019).

However, despite their success, deep learning architectures suffer from important scalability issues, *i.e.*, the actual artificial neural networks (ANN) become unmanageable as the number of features increases.

While most current strategies focus on using more powerful hardware, the approach herein described employs network science strategies to tackle the complexity of ANNs iteratively, that is, at each epoch of the training process.

This work originates in our earlier publication (Mocanu et al. 2018), a promising research avenue to speed up neural network training. There, a new approach called sparse evolutionary training (SET) was defined, in which the acceleration effects obtained by enforcing, in turn, scale freeness, small worldness, and sparsity, during the ANN training process, were explored.

The SET framework firstly initializes an ANN as a sparse weighted Erdős–Rényi graph in which the graph density is fixed ($\epsilon = 20\%$, by default), and assigns weights to edges based on a normal distribution with mean equal to zero. Secondly (*i.e.*, during the revision step), nonzero weights iteratively replace null edges (*i.e.*, links with weight equal to zero) with the twofold goal of reducing the loss on the

Communicated by Yaroslav D. Sergeyev.

✉ Lucia Cavallaro
l.cavallaro@derby.ac.uk

Antonio Liotta
antonio.liotta@unibz.it

[1] University of Derby, Kedleston Road, Derby DE22 1GB, UK

[2] University of Messina, Polo Universitario Annunziata, 98122 Messina, Italy

[3] MIFT Department, University of Messina, 98166 Messina, Italy

[4] Faculty of Computer Science, Free University of Bozen-Bolzano, Bolzano, Italy

Published online: 09 September 2020

training set and to keep the number of connections constant. We should note that the revision step is not only rewiring the links but also re-computing the actual weight of the new links.

The efficiency of this approach has also been recently confirmed by independent researchers, who managed to train a million-node ANN on non-specialized laptops (Liu et al. 2019).

Encouraged by those results, our research has now moved into looking at algorithm tuning parameters to pursue a further acceleration effect, at a negligible accuracy loss. The focus is on the revision stage (determined by the ζ parameter) and on its impact on the training time over epochs. Noteworthy results have been achieved by conducting an in-depth investigation into the optimal tuning of ζ and by providing general guidelines on how to achieve better trade-offs between time and accuracy, as described in Sect. 5.2.

The rest of the paper is organized as follows: Sect. 2 provides the background theories employed in this work. To better position our contribution, Sect. 3 captures the state of the art. Next, Sect. 4 addresses the methodology followed and Sect. 5 shows the results obtained. Finally, Sect. 6 draws the conclusions.

2 Background

This section briefly introduces the main concepts required for understanding this work.

Note that, for the sake of simplicity, the words 'weight' and 'link' are used interchangeably, and only weighted links have been considered. The goal is to demonstrate the effectiveness of the SET approach, aiming at lower revised fraction values, in the context of the multilayer perceptron (MLP) supervised model. MLP is a feed-forward ANN composed by several hidden layers, forming a deep network, as shown in Fig.1. Because of the intra-layer links flow, an MLP can be seen as a fully connected directed graph between the input and output layers.

Supervised learning involves observing several samples of a given dataset, which will be divided into 'training' and 'test' samples. While the former is used to train the neural network, the latter works as a litmus test, as it is compared with the ANN predictions. One can find further details on deep learning in LeCun et al. (2015); Goodfellow et al. (2016).

The construction of a fully connected graph inevitably leads to higher computational costs, as the network grows. To overcome this issue, the SET framework (Mocanu et al. 2018) drew inspiration from human brain models and modelled an ANN topology as a weighted sparse Erdős–Rényi graph in which edges were randomly placed with nodes, according to a fixed probability (Erdős and Rényi 1959; Barabási and Pósfai 2016; Latora et al. 2017).

Like in Mocanu et al. (2018), the edge probability is defined as follows:

$$p\left(W_{ij}^{k}\right) = \frac{\epsilon(n^{k} + n^{k-1})}{n^{k} n^{k-1}}, \tag{1}$$

where $W^{k} \in R^{n^{k-1} \times n^{k}}$ is a sparse weight matrix between the k-th layer and the previous one, $\epsilon \in R^{+}$ is the sparsity parameter, and i, j are a pair of neurons; moreover, n^{k} is the number of neurons in the k-th layer.

As outlined in the previous section, this process led to forcing network sparsity. This stratagem is balanced by introducing the tunable revise fraction parameter ζ, which defines the weights fraction size that needs to be rewired (with a new weight assignment) during the training process.

Indeed, at the end of each epoch, there is a weight adjustment phase. It consists of removing the closest-to-zero links in between layers plus a wider revising range (*i.e.*, ζ). This parameter verifies the correctness of the forced-to-be-zero weights. Subsequently, the framework adds new weights randomly to exactly compensate the removed ones. Thanks to this procedure, the number of links between layers remains constant across different epochs, without isolated neurons (Mocanu et al. 2018).

Herein, the role of ζ is analysed as well as showing how to find a good range of ζ values. Our aim is to strike a good balance between learning speed and accuracy.

3 Related literature

In recent years, ANNs have been widely applied in a broad range of domains such as image classification (He et al. 2016), machine translation (Vaswani et al. 2017), and text to speech (Kalchbrenner et al. 2018).

Previous work proves that the accuracy of an ANN (also known as *model quality*) crucially depends on both the model size (defined as the number of layers and neurons per layers) and the amount of training data (Hestness et al. 2017). Due to these reasons, the amount of resources required to train large ANNs is often prohibitive for real-life applications.

An approach promising to achieve high accuracy even with modest hardware resources is *sparsity* (Gale et al. 2019). An ANN is referred to as sparse when only a subset (hopefully of small size) of the model parameters has a value different from zero. The advantages of sparse networks are obvious. On the one hand, sparse data structures can be used to store matrices associated with the representation of an ANN. On the other hand, most of the matrix multiplications (which constitute the most time expensive stage of neural network computation) can be avoided. Furthermore, previous works (Ullrich et al. 2017; Mocanu et al. 2018) suggested

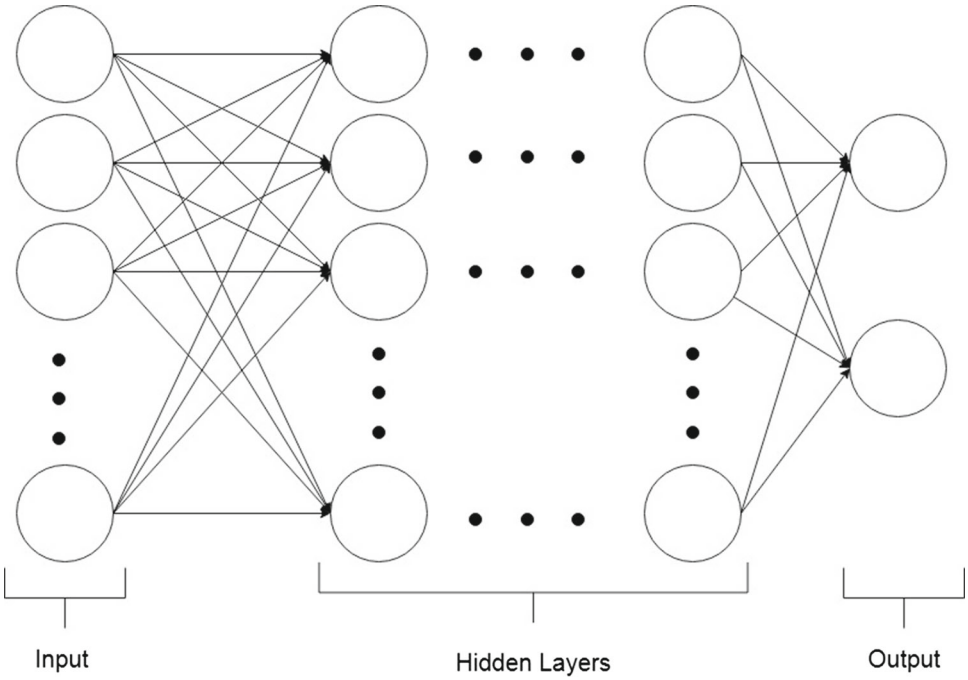


Fig. 1 Example of a generic multilayer perceptron network with more than two hidden layers. Circles represent neurons, and arrows describe the links between layers

that high levels of sparsity do not severely affect the accuracy of an ANN.

This section provides a brief overview of methods used to induce sparse ANNs, by classifying existing methods in two main categories, namely:

1. Methods derived from network science to induce sparse ANNs,
2. Methods derived from ANN regularization to induce sparse ANNs.

3.1 Methods derived from network science to induce sparse ANNs

Some previous papers focus on the interplay between network science and artificial networks (Stier and Granitzer 2019; Mocanu et al. 2018; Bourely et al. 2017). More specifically, they draw inspiration from biological phenomena such as the organization of human brain (Latora et al. 2017; Barabási and Pósfai 2016).

Early studies in network science, in fact, pointed out that real graphs (e.g. social networks describing social ties among members of a community) display important features such as power-law distribution in node degree (Barabási and Pósfai 2016) and the small-world property (Watts and Strogatz 1998). Many authors agree that these properties are likely to exist in many large networked systems one can observe in nature. For instance, in case of biological and neuronal networks, Hilgetag and Goulas (2016) suggested that the neuronal network describing the human brain can be depicted as a globally sparse network with a modular structure.

As a consequence, approaches based on network science consider ANNs as sparse networks whose topological features resemble those of many biological systems and they take advantage from their sparseness to speed up the training stage.

A special mention goes to recent research in Liu et al. (2019), where the authors managed to train a million-node ANN on non-specialized laptops, based on the SET framework that was initially introduce in Mocanu et al. (2018). SET is a training procedure in which connections are pruned on the basis of their magnitude, while other connections are randomly added. The SET algorithm is actually capable of generating ANNs that have sparsely connected layers and, yet, achieve excellent predictive accuracy on real datasets.

Inspired by studies on rewiring in human brain, Bellec et al. (2018) formulated the DEEPR algorithm for training ANNs under connectivity constraints. This algorithm automatically rewires an ANN during the training stage and, to perform such a task, it combines a stochastic gradient descent algorithm with a random walk in the space of parameters to learn.

Bourely et al. (2017) studied to what extent the accuracy of an ANN depends on the density of connections between two consecutive layers. In their approach, they proposed sparse neural network architectures, which derive from random or structured bipartite graphs. Experimental results show that, with a properly chosen topology, sparse neural networks can equal or supersede a fully connected ANN with the same number of nodes and layers in accuracy, with the clear advantage of handling a much smaller parameter space.

Stier and Granitzer (2019) illustrated a procedure to generate ANNs, which derive from artificial graphs. The proposed approach generates a random directed and acyclic graph G according to the Watts-Strogatz (1998) or the Barabási-Albert (2016) models. Nodes in G are then mapped onto layers in an ANN, and some classifiers (such as support vector machines and random forest) are trained to decide if a Watts–Strogatz topology yields a better accuracy than a Barabási–Albert one (or vice versa).

3.2 Methods derived from ANN regularization to induce sparse ANNs

Methods such as L_1 or L_0 regularization, which gained popularity in supervised learning, have been extensively applied to generate compact yet accurate ANNs.

For instance, Srinivas et al. (2017) introduced additional gate variables to efficiently perform model selection. Furthermore, Louizos et al. (2017) described an L_0-norm regularization method, which forces connection weights to become zero. Zero-weight connections are thus pruned, and this is equivalent to induce sparse networks.

The methods above are successful in producing sparse but accurate ANNs; however, they lack explainability. Thus, it is hard to understand why certain architectures are more competitive than others.

It is also interesting to point out that regularization techniques can be viewed as procedures compressing an ANN by deleting unnecessary connections (or, in an equivalent fashion, to select only few parameters). According to Frankle and Carbin (2018), techniques to prune an ANN are effective to uncover sub-networks within an ANN whose initialization made the training process more effective. According to these premises, Frankle and Carbin suggested what they called the *lottery ticket hypothesis*. In other words, dense and randomly initialized ANNs contain sub-networks (called *winning tickets*) that, when trained in isolation, are able to reach the same (or a comparable) test accuracy as the original network, and within a similar number of iterations.

4 Method

Herein we illustrate our research questions and strategy.

To speed up the training process, the investigation relates to the effects drawn by ζ variations during the evolutionary weight phase, at each epoch. The analysis involves a gradual ζ reduction with the goal to provide a guideline on how to find the best ζ values range, to trade-off between speed-up and accuracy loss on different application domains.

In Mocanu et al. (2018), the default revise fraction was set to $\zeta = 0.3$ (i.e. 30% of the revised fraction of nodes) and no further investigations on the sensitivity to ζ were carried out. Unlike in Mocanu et al. (2018)'s research, an in-depth analysis on the revised fraction is herein conducted to understand these effects, particularly how the revise step affects the training when ζ is substantially reduced. In this paper, $\zeta \in [0, 1]$ and $\zeta \in [0\% - 100\%]$ notations are used interchangeably.

Some obvious considerations of this problem are that a shorter execution time and a certain percentage of accuracy loss for smaller values of ζ are expected. Nonetheless, this relationship is bound to be nonlinear; thus, it is crucial to get to quantitative results.

4.1 Dataset and ANN descriptions

The experiments were conducted using well-known datasets, publicly available online[1]:

- LUNG CANCER[2] is a biological dataset composed by features on lung cancer in order to train the ANN to be able to detect them.
- CLL_SUB_111[3] is composed by B-cell chronic lymphocytic leukaemia. This dataset born to profile the five most frequent genomic aberrations (*i.e.*, deletions affecting chromosome bands 13q14, 11q22-q23, 17p13 and 6q21, and gains of genomic material affecting chromosome band 12q13) (Haslinger et al. 2004).
- COIL20[4] is an image dataset used to train ANNs to detect 20 different objects. The images of each object were taken five degrees apart as the object is rotated on a turntable and each object has 72 images. The size of each image is 32×32 pixels, with 256 grey levels per pixel. Thus, each one is represented by a 1024-dimensional vector (Cai et al. 2011, PAMI), (Cai et al. 2011, VLDB).

Both Lung Cancer and CLL_SUB_111 are biological datasets, widely used for their importance in medicine, whereas the COIL20 dataset is a popular images dataset. Further quantitative details are provided in Table 1.

The ANN used is composed of three hidden layers with 3,000 neurons per layer. The activation functions used by default are ReLu for the hidden layers and sigmoid for the output (Table 2).

4.2 Comparison with our previous work

In Mocanu et al. (2018), the goal was to implement the SET algorithm and test it with numerous datasets, on several ANN

[1] http://featureselection.asu.edu/.

[2] https://sites.google.com/site/feipingnie/file/.

[3] https://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE2466.

[4] http://www.cad.zju.edu.cn/home/dengcai/Data/MLData.html.

Table 1 Dataset structures description

Name	Type	Inst. (#)	In. Feat. (#)	Out. C. (#)
Lung cancer	Biological	203	3,312	5
CLL_SUB_111	Biological	111	11,340	3
COIL20	Face image	1440	1024	20

From left: dataset name; dataset type; number of instances, number of input features; number of output classes

Table 2 Artificial neural networks description

Loss function	Batch size (fitting)	Batch size (prediction)	Learning rate	Momentum	Weight decay
MSE	2	1	0.01	0.9	0.0002

It provides information about: the loss function, the batch sizes, the learning rate, the momentum, and the weight decay

types (MLPs, CNN, RBMs), and on different types of tasks (supervised and unsupervised learning). The current study investigates the role of the revise fraction parameter ζ, rather than on the algorithm itself. The aim is to provide a general guideline on finding the best ζ values range to reduce execution time, at a negligible loss of accuracy.

In Cavallaro et al. (2020), a preliminary study on the role of ζ has suggested a negligible accuracy loss, lower fluctuations, and a valuable gain in overall execution time with $\zeta < 0.02$ with the LUNG CANCER dataset. In the present paper, this intuition is analysed on a wider range of datasets to provide stronger justifications for the findings. The most important contribution of our study has been to confirm the effectiveness of the SET framework. Indeed, the random sparseness in ANNs introduced by the SET algorithm is powerful enough even without further fine tuning of weights (*i.e.*, revise fraction) during the training process.

5 Results

This section compares the results obtained by varying the parameter ζ, evaluating the training goodness in terms of the balance between high accuracy reached and short execution time. These topics are treated in Sects. 5.1 and 5.2, respectively. Section 5.3 provides a brief comment on the preferable ζ value, following up from the previous subsections.

For brevity, only the most important outcomes are reported hereafter. The number of epochs was increased from the default value of 100 up to 150 with the aim of finding the ending point of the transient phase. By combining these two tuning parameters (*i.e.*, number of epochs and ζ), we have discovered that, with the datasets herein analysed, the meaningful revise range is $0 \leq \zeta \leq 0.02$.

In particular, Sect. 5.2 shows further investigations in terms of execution time gains, conducted by replicated experiments over ten runs and averaging the obtained results.

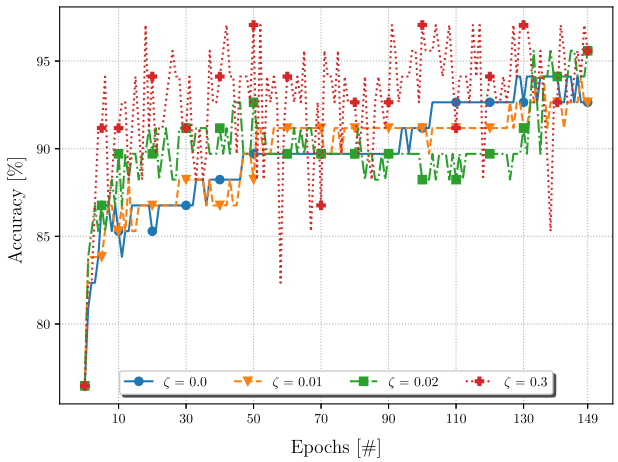
5.1 Accuracy investigation

This section shows the results obtained from the comparative analysis in terms of accuracy improvements over 150 epochs, on the three datasets.

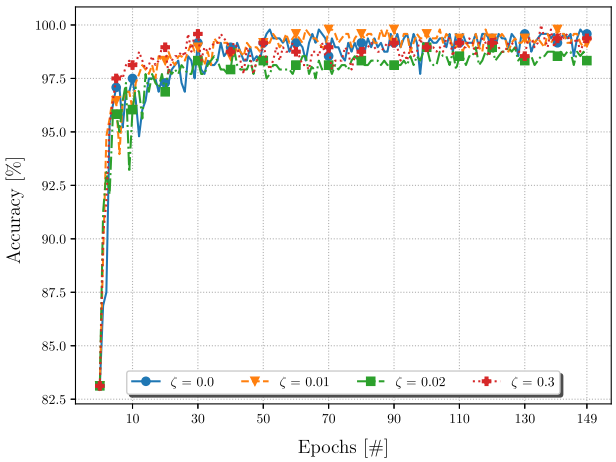
In the LUNG CANCER dataset (Fig. 2a), substantial accuracy fluctuations are present, but there is a no well-defined transient phase for $\zeta > 0.02$. The benchmark value $\zeta = 0.3$ shows an accuracy variation of more than 10% (e.g. accuracy increasing from 82% to 97% at the 60-th epoch and an accuracy from 85% to 95% at the 140th epoch). Note that, since the first 10 epochs are within the settling phase, the significant observations concern the simulation from the 11th epoch. Due to this uncertainty and due to the absence of a transient phase, it is impossible to identify an optimal stopping condition for the algorithm. For instance, at the 60th epoch an accuracy collapse from 97% to 82% was found, followed by an accuracy of 94% at the next epoch.

For a lower revise fraction, *i.e.*, $\zeta \leq 0.02$, an improvement in terms of both stability (*i.e.*, lower fluctuations) and accuracy loss emerges, as expected. In this scenario, defining an exit condition according to the accuracy trend over time is easier. Indeed, despite a higher accuracy loss, the curve stability allows the identification of a gradual accuracy growth over the epochs, with no unexpected sharp drops.

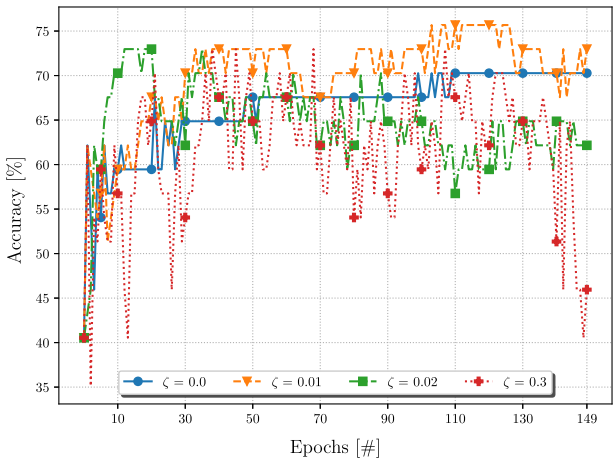
To quantify the amount of accuracy loss, refer to Table 3, which reports both the revise fraction and the highest accuracy reached during the whole simulation, as a percentage. Moreover, mean and confidence interval bounds are provided. From Table 3, it is possible to assert that, on average, the improvement achieved by using a higher revise fraction (as the default one is) has an accuracy gain of just less than 3% (e.g. mean at $\zeta = 0\%$ vs mean at $\zeta = 30\%$) that is a negligible improvement in most of the application domains. This depends on the tolerance level required. For example, if the goal is to achieve an accuracy of at least 90%, then a lower ζ is sufficiently effective. The confidence interval is rather low, given that the fluctuation between the lower and the upper bounds is comprised between 0.8 and 0.9.



(a) *Lung Cancer* dataset.



(b) *COIL20* dataset.



(c) *CLL_SUB_111* dataset.

Fig. 2 Accuracy percentage over 150 epochs varying ζ among [0%, 1%, 2%] plus $\zeta = 30\%$ that is the benchmark value. In particular, $\zeta = 0\%$ with circled markers, $\zeta = 1\%$ has triangular markers, $\zeta = 2\%$ is shown with squared markers, and for $\zeta = 30\%$ cross shape markers have been used

In the COIL20 dataset (Fig. 2b), a short transient phase with no evident improvements among the simulations with different values of ζ emerges. Indeed, there are just small accuracy fluctuations of $\pm 3\%$. These results do not surprise, since improvements achieved through ζ variations also depend on the goodness of the dataset itself, both in terms of its size and in the choice of its features. Table 3 shows that accuracy is always above 98%; thus, even with $\zeta = 0$ the accuracy loss is negligible. Also the confidence interval is lower than 0.3. As the accuracy is continuously increasing over the training epochs, defining a dynamic exit condition is easier in this application domain.

Figure 2c shows the results obtained in CLL_SUB_111 dataset. It is evident that the worse and more unstable approaches among the one considered are both the default one (*i.e.*, $\zeta = 30\%$) and $\zeta = 2\%$.

From Table 3, it is interesting to notice how the accuracy levels are even more stable when using a lower revise fraction (*i.e.*, going from a mean equal to 62.23% in $\zeta = 30\%$ up to 67.14% in $\zeta = 0\%$). The fluctuations compared with the other two datasets are more evident, even when looking at the confidence interval; indeed, it varies from 1.06 (with $\zeta = 0$) up to 2.18 (with $\zeta = 30$), which is larger than the previously analysed one. Because of significant accuracy fluctuations, a possible early exit condition should be considered only with $\zeta = 0$ even at the cost of a slighter higher accuracy loss.

The results obtained so far suggest that there is no need to fine-tune ζ, because the sparsity introduced by the SET algorithm is sufficiently powerful, and only a few links need to be rewired (*i.e.*, $\zeta \leq 0.2$). Apart from the goodness of the datasets themselves (as in COIL20), opting for a lower revise fraction has shown that, on the one hand, the accuracy loss

Table 3 Evaluating parameters varying the revise fraction on datasets considered in a single run with fixed seed

ζ (%)	Max Acc. (%)	Mean (%)	Lower B. (%)	Upper B. (%)
(a) *Lung cancer* dataset				
30%	97.06%	93.13%	92.67%	93.58%
2%	95.59%	90.38%	90.08%	90.68%
1%	94.12%	90.19%	89.84%	90.55%
0%	94.12%	90.21%	89.79%	90.62%
(b) *COIL20* dataset				
30%	100%	98.82%	98.75%	98.90%
2%	98.96%	98.17%	98.08%	98.25%
1%	99.79%	99.09%	98.99%	99.18%
0%	99.79%	98.84 %	98.70%	98.98%
(c) *CLL_SUB_111* dataset				
30%	72.97%	62.23%	61.14%	63.32%
2%	72.97%	65.15%	64.54%	65.76%
1%	75.67%	70.79%	70.15%	71.42%
0%	70.27%	67.14%	66.61%	67.67%

From left: the revise fraction in percentage; the highest accuracy reached during the simulation expressed in percentage; the accuracy mean during the simulation, and the confidence interval bounds. Note that these last three parameters are computed after the first 10 epochs to avoid noise

is sometimes negligible. On the other hand, as it was in the CLL_SUB_111 dataset, the performances are even higher than the ones obtained through the benchmark value. This confirms the hypothesis made in Sect. 5.1 of the goodness of using a randomly sparse ANN topology.

5.2 Execution time investigation

This section shows the comparative analysis conducted among the datasets used, in terms of execution time, over replicated simulations. Ten runs have been averaged, using the default value $\zeta = 0.3$, as benchmark (*i.e.*, $\zeta_{default}$). Note that only the most significant and competitive ζ value has been considered (*i.e.*, $\zeta_0 = 0$). Figure 3 shows the execution time (in seconds) of the same averaged simulations computed on the three datasets.

In both LUNG and CLL_SUB_111 datasets, $\zeta = 0$ is faster than the benchmark value. In particular, in CLL_SUB_111, the execution time is almost 40% faster than the default one and with higher accuracy performances too, as previously asserted in Sect. 5.1. It became less competitive in COIL20. The reason is the same with the results emerged in the accuracy analysis. Indeed, the goodness of the dataset is such as to make insignificant the improvements obtained by varying the revise parameter. Furthermore, the execution time gain between $\zeta = 0$ and $\zeta_{default}$ has been computed among the datasets over ten runs as follows:

$$Gain = 1 - \frac{\zeta_0}{\zeta_{default}} \tag{2}$$

The execution time gain was equal to 0.1370 in LUNG, −0.0052 in COIL20, and 0.3664 in CLL_SUB_111. This means that, except for COIL20, there is an improvement in terms of algorithm performances. Thus, the algorithm became faster using a lower revise fraction. This is even more evident in CLL_SUB_111 as already noticed from Figure 3. On the other hand, the slow down emerged in COIL20 is almost negligible; thus, it may be concluded that for specific types of datasets, there is neither gain nor loss in choosing a lower ζ.

These results confirmed the previous hypothesis of the unnecessary fine-tune ζ process even because, on particular datasets (e.g. COIL20), an in-depth analysis of ζ is profitless. Thus, a relatively low revise fraction has been demonstrated to be a good practice in most of the cases.

5.3 Considerations on the ζ tuning process

In Sects. 5.1 and 5.2, we have described the effects of ζ in terms of accuracy loss and execution time, respectively. This section provides a brief summary of what emerged from those experiments. As largely discussed in the literature, it is unrealistic to try and find a perfect value, which works well in all possible deep learning scenarios *a priori*. The same consideration should be made during the revise fraction tuning. This is why those tests are not aimed at finding the optimal value, which depends on too many variables. Instead, it may be asserted that, from the experiments herein conducted, a relatively low ζ is always a good choice. Indeed, in the datasets analysed the best results have been obtained with $0 \leq \zeta \leq 0.02$. It also important to highlight that because of

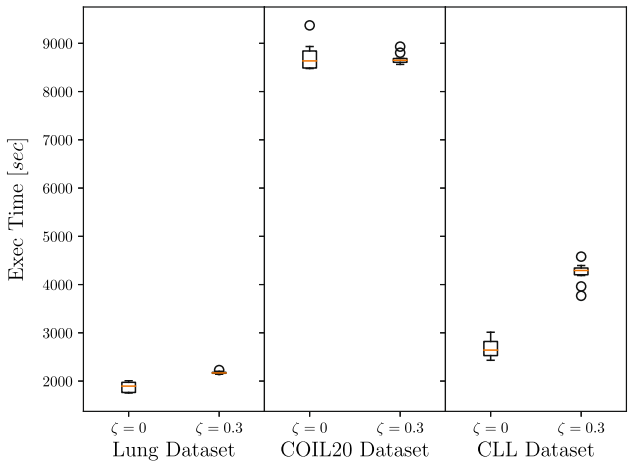


Fig. 3 Execution Time over 10 runs. From left to right, the Lung, COIL20 and CLL datasets are shown

the high non-linearity of the problem itself, more than one ζ value could effectively work, and the process of fine-tuning ζ is an operation that may require more time than the training process itself. This is why this study would provide a good enough range of possible ζ values. Thus, the tests have been conducted on very different datasets to assert that, empirically speaking, in different scenarios $0 \leq \zeta \leq 0.02$ it is sufficient to offer a high accuracy with low fluctuations and, at the same time, faster execution time.

6 Conclusions

In this paper, we moved a step forward from earlier work Mocanu et al. (2018). Not only did our experiments confirm the efficiency arising from training sparse neural networks, but they also managed to further exploit sparsity through a better tuned algorithm, featuring increased speed at a negligible accuracy loss.

The revised fraction goodness is independent from the application domain; thus, a relatively low zeta is always a good practice. Of course, according to the specific scenario considered, the performance may be higher than (or at least equal to) the benchmark value. Yet, it is evident that network science algorithms, by keeping sparsity in ANNs, are a promising direction for accelerating their training processes.

From one side, acting on the revise parameter ζ, accuracy and execution time performances are positively affected. From the other side, it is unrealistic to try and define *a priori* an optimal ζ value, without considering the specific application domain, because of the high non-linearity of the problem. However, through this analysis it is possible to assert that a relatively low ζ is generally sufficient to balance both accuracy loss and execution time. Another strategy could be to sample the dataset in order to manage a lower amount of data and train only that portion of information on which to conduct tests on ζ.

This study paves the way for other works, such as the implementation of dynamic exit conditions to further speed-up the algorithm itself, the development of adaptive algorithms that dynamically tune the parameters, and the study of different distributions for the initial weight assignments.

Acknowledgements We thank Dr Decebal Costantin Mocanu for providing constructive feedback.

Compliance with ethical standards

Conflict of interest All authors declare that they have no conflict of interest.

Human and animal rights This article does not contain any studies with human participants or animals performed by any of the authors.

References

Barabási A-L, Pósfai M (2016) Network science. Cambridge University Press, Cambridge UK

Bellec G, Kappel D, Maass W, Legenstein R (2018) Deep rewiring: training very sparse deep networks. arXiv preprint arXiv:1711.05136

Berman DS, Buczak AL, Chavis JS, Corbett CL (2019) A survey of deep learning methods for cyber security. Information 4:122. https://doi.org/10.3390/info10040122

Bourely A, Boueri JP, Choromonski K (2017) Sparse neural networks topologies. arXiv preprint arXiv:1706.05683

Cai D, He X, Han J, Huang TS (2011) Graph regularized non-negative matrix factorization for data representation. PAMI 33(8):1548–1560

Cai D, He X, Han J (2011) Speed up kernel discriminant analysis. VLDB J 20:21–33. https://doi.org/10.1007/s00778-010-0189-3

Cao C, Liu F, Tan H, Song D, Shu W, Li W, Zhou Y, Bo X, Xie Z (2018) Deep learning and its applications in biomedicine. Genom Proteomics Bioinform 16(1):17–32. https://doi.org/10.1016/j.gpb.2017.07.003

Cavallaro L, Bagdasar O, De Meo P, Fiumara G, Liotta A (2020) Artificial neural networks training acceleration through network science strategies. In: Sergeyev YD, Kvasov DE (eds) Numerical computations: theory and algorithms, NUMTA 2019. Lecture Notes in

Computer Science, Springer, Cham 11974:330–336. https://doi.org/10.1007/978-3-030-40616-5_27
Chen H, Engkvist O, Wang Y, Olivecrona M, Blaschke T (2018) The rise of deep learning in drug discovery. Drug Discov Today 23(6):1241–1250. https://doi.org/10.1016/j.drudis.2018.01.039
Dong Y, Li D (2011) Deep learning and its applications to signal and information processing [exploratory DSP]. IEEE Signal Process Mag 1:145. https://doi.org/10.1109/MSP.2010.939038
Erdős P, Rényi A (1959) On random graphs i. Publ Math-Debr 6:290–297
Frankle J, Carbin M (2018) The Lottery Ticket Hypothesis: Finding Sparse, Trainable Neural Networks. arXiv preprint arXiv:1803.03635
Gale T, Elsen E, Hooker S (2019) the state of sparsity in deep neural networks. arXiv preprint arXiv:1902.09574
Goodfellow I, Bengio Y, Courville A (2016) Deep learning. MIT Press, Cambridge US
Haslinger C, Schweifer N, Stilgenbauer S, Döhner H, Lichter P, Kraut N, Stratowa C, Abseher R (2004) Microarray gene expression profiling of B-cell chronic lymphocytic leukemia subgroups defined by genomic aberrations and VH mutation status. J Clin Oncol 22(19):3937–49. https://doi.org/10.1200/JCO.2004.12.133
Hestness J, Narang S, Ardalani N, Diamos GF, Jun H, Kianinejad H, Patwary MMA, Yang Y, Zhou Y (2017) Deep learning scaling is predictable, empirically. arXiv preprint arXiv:1712.00409
He K, Zhang X, Ren S, Sun J (2016) Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition, CVPR 2016, Las Vegas USA, pp 770–778
Hilgetag CC, Goulas A (2016) Is the brain really a small-world network? Brain Struct Funct 221(4):2361–2366
Hinton G, Deng L, Yu D, Dahl GE, Mohamed A, Jaitly N, Senior A, Vanhoucke V, Nguyen P, Sainath TN, Kingsbury B (2012) Deep neural networks for acoustic modeling in speech recognition. IEEE Signal Process Mag 29:82–97
Kalchbrenner N, Elsen E, Simonyan K, Noury S, Casagrande N, Lockhart E, Stimberg F, van den Oord A, Dieleman S, Kavukcuoglu K (2018) Efficient neural audio synthesis. In: Proceedings of the international conference on machine learning, ICML 2018, Stockholm, pp 2415–2424
Krizhevsky A, Sutskever I, Hinton GE (2017) ImageNet classification with deep convolutional neural networks. Commun ACM 60(6):84–90. https://doi.org/10.1145/3065386
Latora V, Nicosia V, Russo G (2017) Complex networks: principles, methods and applications. Cambridge University Press, Cambridge UK
LeCun Y, Bengio Y, Hinton G (2015) Deep learning. Nature 521:436–444. https://doi.org/10.1038/nature14539
Liu S, Mocanu DC, Matavalam ARR, Pei Y, Pechenizkiy M (2019) Sparse evolutionary Deep Learning with over one million artificial neurons on commodity hardware. ArXiv, arXiv:1901.09181
Louizos C, Welling M, Kingma DP (2017) Learning sparse neural networks through L_0 Regularization. arXiv preprint arXiv:1712.01312
Mocanu DC, Mocanu E, Stone P, Nguyen PH, Gibescu M, Liotta A (2018) Scalable training of artificial neural networks with adaptive sparse connectivity inspired by network science. Nat Commun 9:2383. https://doi.org/10.1038/s41467-018-04316-3
Ruano-Ordás D, Yevseyeva I, Fernandes VB, Méndez JR, Emmerich MTM (2019) Improving the drug discovery process by using multiple classifier systems. Expert Syst Appl 121:292–303. https://doi.org/10.1016/j.eswa.2018.12.032
Srinivas S, Subramanya A, Babu RV (2017) Training sparse neural networks. In: Proceedings of the IEEE conference on computer vision and pattern recognition workshops, Honolulu, pp 455–462. https://doi.org/10.1109/CVPRW.2017.61
Stier J, Granitzer M (2019) Structural analysis of sparse neural networks. Procedia Comput Sci 159:107–116
Ullrich K, Meeds E, Welling M (2017) Soft weight-sharing for neural network compression. arXiv preprint arXiv:1702.04008
Vaswani A, Shazeer N, Parmar N, Uszkoreit J, Jones L, Gomez AN, Kaiser L, Polosukhin I (2017) Attention is all you need. In: Proceedings of the annual conference on neural information processing systems, Long Beach, USA, pp 6000–6010
Watts DJ, Strogatz SH (1998) Collective dynamics of 'small-world' networks. Nature 393:440–442